Amendments to the Claims

Listing of the Claims:

1. (currently amended) A computer-implemented method for analyzing a query, the query including one or more conditions and one or more sub-queries, the conditions including one or more connecting conditions that introduce the sub-query in the query, each of the sub-queries including zero or more conditions, the method including:

determining the satisfiability of the query, including:

determining the satisfiability of the connecting conditions; and

determining the satisfiability of the conditions in the sub-queries.

2. (original) The method of claim 1, where determining the satisfiability of the query further includes determining the satisfiability of all other conditions.

3. (original) The method of claim 2, where determining the satisfiability of the conditions includes:

creating and populating a global conditions set; and

determining the satisfiability of the global conditions set.

4. (original) The method of claim 3, where the query includes a clause of the form (X CC (SELECT Y FROM T)), where CC is a connecting condition, X and Y are variables or columns, T is a set of one or more tables or views,, and where populating the global conditions set includes:

if CC is "IN," adding (X=Y) to the global conditions set;

if CC is "NOT IN," adding (X<>Y) to the global conditions set; and

if CC includes arithmetic comparison COMP, adding (X COMP Y) to the global conditions set.

5. (original) The method of claim 3, where the query includes a clause of the form (CC (SELECT Y FROM T WHERE R)), where CC is a connecting condition, Y is a variables or a column, T is a set of one or more tables or views, and R is a set of one or more conditions, and where populating the global conditions set includes adding R to the global conditions set.

6. (original) The method of claim 3, where determining the satisfiability of the global conditions set includes:

converting the form of the conditions in the global conditions set to less-than-or-equal-to conditions;

creating a map M of the less-than-or-equal-to conditions;

finding the shortest path between all nodes in M; and

determining if M has a negative cycle and, if it does, returning that the query is not satisfiable.

7. (original) The method of claim 6, where creating the map M of the conditions in the global conditions set includes:

creating a node for each of the variables in the conditions;

creating a node for 0;

creating a directed edge from a node representing a first variable, S, to a node representing a second variable, T, with a cost, C, for conditions of the form (S<=T+C);

creating a directed edge from a node representing a first variable, S, to the 0 node, with cost C, for conditions of the form (S<=0+C); and

creating a directed edge from the 0 node to a node representing a first variable, S, with cost C, for conditions of the form (0<=X+C).

8. (original) The method of claim 6, where finding the shortest path between all nodes in M includes

running the Floyd-Warshall Shortest Path Algorithm against M.

9. (original) The method of claim 6, where determining if M has a negative cycle includes determining if M includes a negative cost edge from a node to itself.

10. (original) The method of claim 6, where analyzing the query further includes:

determining the transitive closure of the conditions and, if necessary, modifying the conditions.

11. (original) The method of claim 10, where the query includes an outer query block and an inner query block, and where determining the transitive closure of the conditions and modifying the conditions includes:

determining the transitive closure of conditions in the outer query block and, if necessary, modifying the conditions in the outer query block and

determining the transitive closure of conditions in the inner query block and, if necessary, modifying the conditions in the inner query block.

12. (original) The method of claim 10, where determining the transitive closure of the conditions includes:

creating and populating a global conditions set including conditions from the outer query block and the inner query block; and

determining the transitive closure of the global conditions set.

13. (original) The method of claim 12, where the transitive closure includes one or more transitive closure conditions, and where modifying the conditions to achieve transitive includes:

for each transitive closure condition of the form (COL COMP C), where COL is a column, COMP is a comparison, and C is a constant:

if COL appears in the outer query block, adding the transitive closure condition to the outer query block; and

if COL appears in the inner query block, adding the transitive closure condition to the inner query block.

14. (currently amended) A computer-implemented method for analyzing a query, the query including one or more conditions of the form (X+Y OP C), where X and Y are variables, C is a constant, and OP is an operator, the method including:

determining the satisfiability of the query, including:

determining the satisfiability of the one or more conditions of the form (X+Y OP C).

15. (original) The method of claim 14, where a negation of OP is represented by the operator OP', and where determining the satisfiability of the query includes:

assigning conditions of the form (X OP Y+C) to a set S1;

assigning condition of the form (X+Y OP C) to a set S2;

assigning conditions of the form (X OP C) to a set S3;

replacing each conditions in set S2 with two conditions in the form (Y OP -X+C) and (X OP -Y+C);

if -X is present in set S2:

for each condition in set S3:

adding a condition of the form (-X OP' -C) to set S3; and

determining the satisfiability of the group of conditions (S1 UNION S2 UNION S3).

16. (original) The method of claim 15, where determining the satisfiability of the group of conditions includes:

converting the conditions to less-than-or-equal-to conditions;

creating a map M of the less-than-or-equal-to conditions;

finding the shortest path between all nodes in M; and

determining if M has a negative cycle and, if it does, returning that the conditions are not satisfiable.

17. (original) The method of claim 16, where creating the map M of the conditions in the global conditions set includes:

creating a node for each of the variables in the conditions, including creating separate nodes for variables with opposite signs;

creating a node for 0;

creating a directed edge from a node representing a first variable, S, to a node representing a second variable, T, with a cost, C, for conditions of the form (S<=T+C);

creating a directed edge from a node representing a first variable, S, to the 0 node, with cost C, for conditions of the form (S<=0+C); and

creating a directed edge from the 0 node to a node representing a first variable, S, with cost C, for conditions of the form (0<=X+C).

18. (original) The method of claim 14, where analyzing the query further includes:

determining the transitive closure of the conditions and, if necessary, modifying the conditions.

19. (original) A computer-implemented method for analyzing a query, the query including one or more conditions and one or more sub-queries, the conditions including one or more connecting conditions that introduce the sub-query in the query, each of the sub-queries including zero or more conditions, the method including:

creating a Global Conditions set including one or more conditions representing one or more connecting conditions.

20. (original) A computer-implemented method for analyzing a query, the query including one or more conditions and one or more sub-queries, the conditions including one or more connecting conditions that introduce the sub-query in the query, each of the sub-queries including zero or more conditions, the method including:

creating a Transitive Closure set of conditions based on one or more connecting conditions.

21. (original) A computer program, stored on a tangible storage medium, for use in analyzing one or more database queries, each query including one or more conditions and one or more sub-queries, the conditions including one or more connecting conditions that introduce the sub-query in the query, each of the sub-queries including zero or more conditions, the computer program including executable instructions that cause a computer to:

determine the satisfiability of the query, including executable instructions that cause the computer to:

determine the satisfiability of the connecting conditions; and

determine the satisfiability of the conditions in the sub-queries.

22. (original) The computer program of claim 21, where the executable instructions for determining the satisfiability of the query cause the computer to determine the satisfiability of all other conditions.

23. (original) The computer program of claim 22, where the executable instructions for determining the satisfiability of the query cause the computer to:

create and populate a global conditions set; and

determine the satisfiability of the global conditions set.

24. (original) The computer program of claim 23, where the query includes a clause of the form (X CC (SELECT Y FROM T)), where CC is a connecting condition, X and Y are variables or columns, T is a set of one or more tables or views, and where the executable instructions for populating the global conditions cause the computer to:

if CC is "IN," add (X=Y) to the global conditions set;

if CC is "NOT IN," add (X<>Y) to the global conditions set; and

if CC includes arithmetic comparison COMP, add (X COMP Y) to the global conditions set.

25. (original) The computer program of claim 23, where the query includes a clause of the form (CC (SELECT Y FROM T WHERE R)), where CC is a connecting condition, Y is a variable or a column, T is a set of one or more tables or views, and R is a set of one or more conditions, and where the executable instructions for populating the global conditions set cause the computer to add R to the global conditions set.

26. (original) The computer program of claim 23, where the executable instructions for determining the satisfiability of the global conditions set cause the computer to:

convert the form of the conditions in the global conditions set to less-than-or-equal-to conditions;

create a map M of the less-than-or-equal-to conditions;

find the shortest path between all nodes in M; and

determine if M has a negative cycle and, if it does, return that the query is not satisfiable.

27. (original) The computer program of claim 26, where the executable instructions for creating the map M of the conditions in the global conditions set cause the computer to:

create a node for each of the variables in the conditions;

create a node for 0;

create a directed edge from a node representing a first variable, S, to a node representing a second variable, T, with a cost, C, for conditions of the form (S<=T+C);

create a directed edge from a node representing a first variable, S, to the 0 node, with cost C, for conditions of the form (S<=0+C); and

create a directed edge from the 0 node to a node representing a first variable, S, with cost C, for conditions of the form (0<=X+C).

28. (original) The computer program of claim 26, where the executable instructions for finding the shortest path between all nodes in M cause the computer to:

run the Floyd-Warshall Shortest Path Algorithm against M.

29. (original) The computer program of claim 26, where the executable instructions for determining if M has a negative cycle cause the computer to:

determine if M includes a negative cost edge from a node to itself.

30. (original) The computer program of claim 21, where the executable instructions for analyzing the query cause the computer to:

determine the transitive closure of the conditions and, if necessary, modifying the conditions.

31. (original) The computer program of claim 30, where the query includes an outer query block and an inner query block, and where the executable instructions for determining the transitive closure of the conditions and modifying the conditions cause the computer to:

determine the transitive closure of conditions in the outer query block and, if necessary, modify the conditions in the outer query block and

determine the transitive closure of conditions in the inner query block and, if necessary, modify the conditions in the inner query block.

32. (original) The computer program of claim 30, where the executable instructions for determining the transitive closure of the conditions cause the computer to:

create and populate a global conditions set including conditions from the outer query block and the inner query block; and

determine the transitive closure of the global conditions set.

33. (original) The computer program of claim 23, where the transitive closure includes one or more transitive closure conditions, and where the executable instructions for modifying the conditions to achieve transitive cause the computer to:

for each transitive closure condition of the form (COL COMP C), where COL is a column, COMP is a comparison, and C is a constant:

if COL appears in the outer query block, add the transitive closure condition to the outer query block; and

if COL appears in the inner query block, add the transitive closure condition to the inner query block.

34. (original) A computer program, stored on a tangible storage medium, for use in analyzing one or more database queries, each query including one or more conditions of the form (X+Y OP C), where X and Y are variables or columns, C is a constant, and OP is an operator, the executable instructions that cause a computer to:

determine the satisfiability of the query, including executable instruction that cause the computer to:

determine the satisfiability of the one or more conditions of the form (X+Y OP C).

35. (original) The computer program of claim 34, where a negation of OP is represented by the operator OP', and where the executable instructions for determining the satisfiability of the query cause the computer to:

assign conditions of the form (X OP Y+C) to a set S1;

assign condition of the form (X+Y OP C) to a set S2;

assign conditions of the form (X OP C) to a set S3;

replace each conditions in set S2 with two conditions in the form (Y OP -X+C) and (X OP -Y+C);

if -X is present in set S2:

for each condition in set S3:

add a condition of the form (-X OP' -C) to set S3; and

determine the satisfiability of the group of conditions (S1 UNION S2 UNION S3).

36. (original) The computer program of claim 35, where the executable instructions for determining the satisfiability of the group of conditions cause the computer to:

convert the conditions to less-than-or-equal-to conditions;

create a map M of the less-than-or-equal-to conditions;

find the shortest path between all nodes in M; and

determine if M has a negative cycle and, if it does, return that the conditions are not satisfiable.

37. (original) The computer program of claim 36, where the executable instruction for creating the map M of the conditions in the global conditions set cause the computer to:

create a node for each of the variables in the conditions, including creating separate nodes for variables with opposite signs;

create a node for 0;

create a directed edge from a node representing a first variable, S, to a node representing a second variable, T, with a cost, C, for conditions of the form (S<=T+C);

create a directed edge from a node representing a first variable, S, to the 0 node, with cost C, for conditions of the form (S<=0+C); and

create a directed edge from the 0 node to a node representing a first variable, S, with cost C, for conditions of the form (0<=X+C).

38. (original) The computer program of claim 34, where the executable instruction for analyzing the query further cause the computer to:

determine the transitive closure of the conditions and, if necessary, modifying the conditions.

39. (original) A database system including:

a massively parallel processing system including:

one or more nodes;

a plurality of CPUs, each of the one or more nodes providing access to one or more CPUs;

a plurality of data storage facilities each of the one or more CPUs providing access to one or more data storage facilities;

a process for execution on the massively parallel processing system for analyzing one or more database queries, each query including one or more conditions and one or more sub-queries, the conditions including one or more connecting conditions that introduce the sub-query in the query, each of the sub-queries including zero or more conditions, the process including:

determining the satisfiability of the query, including:

determining the satisfiability of the connecting conditions; and

determining the satisfiability of the conditions in the sub-queries.

40. (original) The database system of claim 39, where determining the satisfiability of the query includes determining the satisfiability of all other conditions.

41. (original) The database system of claim 40, where determining the satisfiability of the conditions includes:

creating and populating a global conditions set; and

determining the satisfiability of the global conditions set.

42. (original) The database system of claim 41, where the query includes a clause of the form (X CC (SELECT Y FROM T)), where CC is a connecting condition, X and Y are variables or columns T is a set of one or more tables or views, and where populating the global conditions set includes:

if CC is "IN," adding (X=Y) to the global conditions set;

if CC is "NOT IN," adding (X<>Y) to the global conditions set; and

if CC includes arithmetic comparison COMP, adding (X COMP Y) to the global conditions set.

43. (original) The database system of claim 41, where the query includes a clause of the form (CC (SELECT Y FROM T WHERE R)), where CC is a connecting condition, Y is a variable or a column, T is a set of one or more tables or views, and R is a set of one or more conditions, and where populating the global conditions set includes adding R to the global conditions set.

44. (original) The database system of claim 41, where determining the satisfiability of the global conditions set includes:

converting the form of the conditions in the global conditions set to less-than-or-equal-to conditions;

creating a map M of the less-than-or-equal-to conditions;

finding the shortest path between all nodes in M; and

determining if M has a negative cycle and, if it does, returning that the query is not satisfiable.

45. (original) The database system of claim 44, where creating the map M of the conditions in the global conditions set includes:

creating a node for each of the variables in the conditions;

creating a node for 0;

creating a directed edge from a node representing a first variable, S, to a node representing a second variable, T, with a cost, C, for conditions of the form (S<=T+C);

creating a directed edge from a node representing a first variable, S, to the 0 node, with cost C, for conditions of the form (S<=0+C); and

creating a directed edge from the 0 node to a node representing a first variable, S, with cost C, for conditions of the form (0<=X+C).

46. (original) The database system of claim 44, where finding the shortest path between all nodes in M includes

running the Floyd-Warshall Shortest Path Algorithm against M.

47. (original) The database system of claim 44, where determining if M has a negative cycle includes

determining if M includes a negative cost edge from a node to itself.

48. (original) The database system of claim 39, where the process further includes:

determining the transitive closure of the conditions and, if necessary, modifying the conditions.

49. (original) The database system of claim 48, where the query includes an outer query block and an inner query block, and where determining the transitive closure of the conditions and modifying the conditions includes:

determining the transitive closure of conditions in the outer query block and, if necessary, modifying the conditions in the outer query block and

determining the transitive closure of conditions in the inner query block and, if necessary, modifying the conditions in the inner query block.

50. (original) The database system of claim 48, where determining the transitive closure of the conditions includes:

creating and populating a global conditions set including conditions from the outer query block and the inner query block; and

determining the transitive closure of the global conditions set.

51. (original) The database system of claim 50, where the transitive closure includes one or more transitive closure conditions, and where modifying the conditions to achieve transitive includes:

for each transitive closure condition of the form (COL COMP C), where COL is a column, COMP is a comparison, and C is a constant:

if COL appears in the outer query block, adding the transitive closure condition to the outer query block; and

if COL appears in the inner query block, adding the transitive closure condition to the inner query block.

52. (original) A database system including:

a massively parallel processing system including:

one or more nodes;

a plurality of CPUs, each of the one or more nodes providing access to one or more CPUs;

a plurality of data storage facilities each of the one or more CPUs providing access to one or more data storage facilities;

a process for execution on the massively parallel processing system for analyzing one or more database queries, each query including one or more conditions of the form (X+Y OP C), where X and Y are variables, C is a constant, and OP is an operator, the process including:

determining the satisfiability of the query, including:

determining the satisfiability of the one or more conditions of the form (X+Y OP C).

53. (original) The database system of claim 52, where a negation of OP is represented by the operator OP', and where determining the satisfiability of the query includes:

assigning conditions of the form (X OP Y+C) to a set S1;

assigning condition of the form (X+Y OP C) to a set S2;

assigning conditions of the form (X OP C) to a set S3;

replacing each conditions in set S2 with two conditions in the form (Y OP -X+C) and (X OP -Y+C);

if -X is present in set S2:

for each condition in set S3:

adding a condition of the form (-X OP' -C) to set S3; and

determining the satisfiability of the group of conditions (S1 UNION S2 UNION S3).

54. (original) The database system of claim 53, where determining the satisfiability of the group of conditions includes:

converting the conditions to less-than-or-equal-to conditions;

creating a map M of the less-than-or-equal-to conditions;

finding the shortest path between all nodes in M; and

determining if M has a negative cycle and, if it does, returning that the conditions are not satisfiable.

55. (original) The database system of claim 54, where creating the map M of the conditions in the global conditions set includes:

creating a node for each of the variables in the conditions, including creating separate nodes for variables with opposite signs;

creating a node for 0;

creating a directed edge from a node representing a first variable, S, to a node representing a second variable, T, with a cost, C, for conditions of the form (S<=T+C);

creating a directed edge from a node representing a first variable, S, to the 0 node, with cost C, for conditions of the form (S<=0+C); and

creating a directed edge from the 0 node to a node representing a first variable, S, with cost C, for conditions of the form (0<=X+C).

56. (original) The database system of claim 52, where the process further includes:

determining the transitive closure of the conditions and, if necessary, modifying the conditions.